



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

**Институт информационных технологий (ИИТ)
Кафедра цифровой трансформации (ЦТ)**

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ
по дисциплине «Разработка баз данных»

Практическое занятие № 5

Студенты группы *ИКБО-42-23 Голев С.С.*

(подпись)

Ассистент *Морозов Д.В.*

(подпись)

Отчет представлен «___»____ 2025 г.

Москва 2025 г.

СОДЕРЖАНИЕ

ЗАДАНИЕ.....	3
ВЫПОЛНЕНИЕ ЗАДАНИЯ.....	6

ЗАДАНИЕ

Задание №1: создание модифицируемого представления Для вашей базы данных создать простое модифицируемое представление, которое отбирает строки из одной таблицы по определенному критерию.

Например, для БД «Аптека» можно создать представление, отображающее лекарства только от одного производителя «Bayer AG»).

Задание №2: модификация данных через представление

Продемонстрировать возможность изменения данных в базовой таблице через представление, созданное в Задании №1. Для этого необходимо выполнить два запроса:

1. Добавить новую запись с помощью оператора INSERT.
2. Удалить существующую запись с помощью оператора DELETE.

Задание №3: создание немодифицируемого аналитического представления Для вашей базы данных создать единое немодифицируемое представление для аналитических целей.

Представление должно объединять данные как минимум из двух таблиц и содержать агрегирующие функции (COUNT, SUM, AVG и т.д.) и группировку (GROUP BY).

Например, для БД «Аптека» такое представление могло бы для каждого производителя выводить общее количество наименований лекарств и их среднюю цену.

Задание №4: использование аналитического представления в запросах

Написать SELECT-запрос, который использует созданное в Задании №3 аналитическое представление в качестве источника данных для дальнейшей фильтрации или анализа.

Например, можно отобрать производителей, у которых средняя цена на продукцию превышает определенное значение.

Задание №5: Создание и обновление материализованного представления

1. Создать материализованное представление для ускорения выполнения ресурсоемкого аналитического запроса.

Например, для БД «Аптека» можно создать представление, которое заранее рассчитывает общую сумму продаж для каждого покупателя.

2. Продемонстрировать процесс обновления данных в представлении с помощью команды REFRESH MATERIALIZED VIEW viewName;.

Задание №6: разработка пользовательской функции для аналитических вычислений

1. Разработать пользовательскую функцию, которая инкапсулирует комплексный аналитический расчет. Функция должна принимать на вход идентификатор (например, manufacturer_id) и возвращать одно скалярное значение (например, общую сумму продаж продукции данного производителя), вычисленное на основе соединения нескольких таблиц и применения агрегатных функций.

2. Продемонстрировать вызов функции в составе SELECT-запроса.

Задание №7: разработка хранимой процедуры для выполнения сложной операции. Разработайте хранимую процедуру, которая выполняет безопасную операцию по изменению данных. Процедура должна принимать на вход ID какой-либо записи и числовое значение (например, количество). Внутри процедуры необходимо проверить, достаточно ли текущего значения в числовом поле одной таблицы для выполнения операции.

- Если да – уменьшите это значение и добавьте новую запись в другую, связанную таблицу.
- Если нет – операция должна полностью прерваться, не внося никаких изменений в данные.

Для сообщения о результате используйте выходной параметр, который вернёт статус успеха или неудачи.

Задание №8: демонстрация вызова хранимой процедуры

Привести два примера вызова процедуры, созданной в Задании №7:

- Успешный вызов, который добавляет в вашу базу данных уникальную

запись.

- Неудачный вызов, который демонстрирует срабатывание реализованной проверки целостности и возврат пользовательской ошибки.

Каждый SQL-запрос сопроводить комментарием, объясняющим его назначение и логику работы с учетом специфики вашей базы данных.

ВЫПОЛНЕНИЕ ЗАДАНИЯ

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor window containing the following SQL code:

```
CREATE OR REPLACE VIEW hardware_innovations_products AS
SELECT
    id_rtl,
    title,
    id_supplier,
    description,
    price
FROM rtl
WHERE id_supplier = 21;
```

In the bottom-right pane, there is a statistics window titled "Статистика 1" (Statistics 1) with the following data:

Name	Value
Updated Rows	0
Execute time	0.025s
Start time	Sat Nov 08 14:26:16 MSK 2025
Finish time	Sat Nov 08 14:26:16 MSK 2025
Query	CREATE OR REPLACE VIEW hardware_innovations_products AS SELECT id_rtl, title, id_supplier, description, price FROM rtl WHERE id_supplier = 21;

Рисунок 1 – создание модифицированного представления

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor window containing the following SQL code:

```
INSERT INTO hardware_innovations_products (title, id_supplier, description, price)
VALUES ('Smart Plug SP1', 21, 'Wi-Fi smart plug with energy monitoring', 59);
SELECT * FROM rtl WHERE title = 'Smart Plug SP1';
DELETE FROM hardware_innovations_products
WHERE title = 'Smart Plug SP1';
SELECT * FROM rtl WHERE title = 'Smart Plug SP1';
```

In the bottom-right pane, there is a statistics window titled "Статистика 1" (Statistics 1) with the following data:

Name	Value
Updated Rows	0
Execute time	0.029s
Start time	Sat Nov 08 14:26:59 MSK 2025
Finish time	Sat Nov 08 14:26:59 MSK 2025
Query	DELETE FROM hardware_innovations_products WHERE title = 'Smart Plug SP1'

Рисунок 2 – модификация данных представления

```
CREATE OR REPLACE VIEW supplier_summary AS
SELECT
    s.org_title AS supplier_name,
    COUNT(r.id_rtl) AS total_products,
    ROUND(AVG(r.price), 2) AS average_price
FROM supplier s
LEFT JOIN rtl r ON s.id_supplier = r.id_supplier
GROUP BY s.org_title;
```

Статистика 1

Name	Value
Updated Rows	0
Execute time	0.014s
Start time	Sat Nov 08 14:27:17 MSK 2025
Finish time	Sat Nov 08 14:27:17 MSK 2025
Query	CREATE OR REPLACE VIEW supplier_summary AS SELECT s.org_title AS supplier_name, COUNT(r.id_rtl) AS total_products, ROUND(AVG(r.price), 2) AS average_price FROM supplier s LEFT JOIN rtl r ON s.id_supplier = r.id_supplier GROUP BY s.org_title

Рисунок 3 – создание немодифицированного аналитического представления

```
SELECT supplier_name, total_products, average_price
FROM supplier_summary
WHERE average_price > 80
ORDER BY average_price DESC;
```

supplier_summary 1

supplier_name	total_products	average_price
Tech Components LLC	1	159
Hardware Innovations Inc.	1	129

Обновить Save Cancel Экспорт данных ... 200 2 2 строки получено - 0.011s, 2025-11-08 в 14:27:34

Рисунок 4 – использование аналитического представления в вопросах

```

CREATE MATERIALIZED VIEW total_spent_by_client AS
SELECT
    c.id_client,
    c.name,
    SUM(o.total) AS total_spent
FROM client c
JOIN ordering o ON c.id_client = o.id_client
GROUP BY c.id_client, c.name;

SELECT * FROM total_spent_by_client;

REFRESH MATERIALIZED VIEW total_spent_by_client;

```

total_spent_by_client x Статистика 1

Выводить только записи с изменениями

	id_client	name	total_spent
1	201	Alice Johnson	199,99
2	205	Elena Rossi	45,75
3	206	Farid Khan	599
4	204	David Chen	1 299
5	203	Carla Gómez	89
6	202	Boris Petrov	349,5

Обновить Save Cancel Экспорт данных 200 6 6 строк получено - 0,019s, 2025-11-08 в 14:28:14

Рисунок 5 – Создание и обновление материализованного представления

```

CREATE OR REPLACE FUNCTION get_client_total_spent(p_client_id INT)
RETURNS DECIMAL(12, 2) AS $$
DECLARE
    total_spent DECIMAL(12, 2);
    client_exists BOOLEAN;
BEGIN
    SELECT EXISTS(SELECT 1 FROM client WHERE id_client = p_client_id)
    INTO client_exists;

    IF NOT client_exists THEN
        RAISE EXCEPTION 'Client with ID % does not exist.', p_client_id;
    END IF;

    SELECT COALESCE(SUM(total), 0.00)
    INTO total_spent
    FROM ordering
    WHERE id_client = p_client_id;

    RETURN total_spent;
END;
$$ LANGUAGE plpgsql;

SELECT name, get_client_total_spent(id_client) AS total_spent
FROM client

```

client 1 x Статистика 1

Выводить только записи с изменениями

	name	total_spent
1	David Chen	1 299
2	Farid Khan	599
3	Boris Petrov	349,5
4	Alice Johnson	297,99
5	Carla Gómez	89
6	Elena Rossi	45,75
7	Ivan Ivanov	0

Обновить Save Cancel Экспорт данных 200 7 7 строк получено - 0,080s, 2025-11-08 в 14:28:28

Рисунок 6 – Разработка пользовательской функции для аналитических вычислений

```

CREATE OR REPLACE PROCEDURE process_order(
    p_rtl_id INT,
    p_client_id INT,
    p_employee_id INT,
    p_quantity INT,
    OUT p_success BOOLEAN,
    OUT p_message TEXT
)
LANGUAGE plpgsql AS $$

DECLARE
    current_price DECIMAL(10,2);
    total_sum DECIMAL(10,2);
    product_exists BOOLEAN;
BEGIN
    SELECT EXISTS(SELECT 1 FROM rtl WHERE id_rtl = p_rtl_id)
    INTO product_exists;

    IF NOT product_exists THEN
        p_success := FALSE;
        p_message := 'Ошибка: товар с ID ' || p_rtl_id || ' не найден.';
        RETURN;
    END IF;

    SELECT price INTO current_price FROM rtl WHERE id_rtl = p_rtl_id;

    total_sum := current_price * p_quantity;

    INSERT INTO ordering (id, id_ordering, id_employee, id_address, id_client, total, date, status, delivery_method)
    VALUES (
        (SELECT MAX(id)+1 FROM ordering),
        (SELECT MAX(id_ordering)+1 FROM ordering),
        p_employee_id,
        (SELECT id_address FROM client_address WHERE id_client = p_client_id LIMIT 1),
        p_client_id,
        total_sum,
        CURRENT_DATE,
        'paid',
        'courier'
    );

    p_success := TRUE;
    p_message := 'Заказ успешно оформлен на сумму ' || total_sum;
END;
$$;

```

Рисунок 7 – Разработка хранимой процедуры для выполнения сложной операции

```

DO $$ 
DECLARE
    v_success BOOLEAN;
    v_message TEXT;
BEGIN
    CALL process_order(801, 201, 501, 2, v_success, v_message);
    RAISE NOTICE '%', v_message;
END;
$$;

DO $$ 
DECLARE
    _success BOOLEAN;
    v_message TEXT;
BEGIN
    CALL process_order(999, 201, 501, 2, _success, v_message);
    RAISE NOTICE '%', v_message;
END;
$$;

```

Name	Value
Updated Rows	0
Execute time	0.013s
Start time	Sat Nov 08 14:29:09 MSK 2025
Finish time	Sat Nov 08 14:29:09 MSK 2025
Query	DO \$\$ DECLARE v_success BOOLEAN; v_message TEXT; BEGIN CALL process_order(801, 201, 501, 2, v_success, v_message); RAISE NOTICE '%', v_message; END; \$\$; DO \$\$ DECLARE _success BOOLEAN; v_message TEXT; BEGIN CALL process_order(999, 201, 501, 2, _success, v_message); RAISE NOTICE '%', v_message; END; \$\$;

Рисунок 8 – Демонстрация вызова хранимой процедуры